

Learning Better Word Embedding by Asymmetric Low-Rank Projection of Knowledge Graph

Fei Tian

University of Science and Technology of China
tianfei@mail.ustc.edu.cn

Bin Gao

Microsoft Research
bingao@microsoft.com

Enhong Chen

University of Science and Technology of China
cheneh@ustc.edu.cn

Tie-Yan Liu

Microsoft Research
tyliu@microsoft.com

Abstract

Word embedding, which refers to low-dimensional dense vector representations of natural words, has demonstrated its power in many natural language processing tasks. However, it may suffer from the inaccurate and incomplete information contained in the free text corpus as training data. To tackle this challenge, there have been quite a few works that leverage knowledge graphs as an additional information source to improve the quality of word embedding. Although these works have achieved certain success, they have neglected some important facts about knowledge graphs: (i) many relationships in knowledge graphs are *many-to-one*, *one-to-many* or even *many-to-many*, rather than simply *one-to-one*; (ii) most head entities and tail entities in knowledge graphs come from very different semantic spaces. To address these issues, in this paper, we propose a new algorithm named ProjectNet. ProjectNet models the relationships between head and tail entities after transforming them with different low-rank projection matrices. The low-rank projection can allow non *one-to-one* relationships between entities, while different projection matrices for head and tail entities allow them to originate in different semantic spaces. The experimental results demonstrate that ProjectNet yields more accurate word embedding than previous works, thus leads to clear improvements in various natural language processing tasks.

1 Introduction

In recent years, the research on word embedding (or distributed word representations) has made promising progresses in many natural language processing tasks [1; 7; 11; 15; 16; 18]. Different from traditional one-hot discrete representations of words, word embedding vectors are dense, continuous, and low-dimensional. They are usually trained with neural networks on a large-scale free text corpus, such as Wikipedia, news articles, and web pages, in an unsupervised manner.

While word embedding has demonstrated its power in many circumstances, it is gradually recognized that conven-

tional word embedding techniques may suffer from the incomplete and inaccurate information contained in the free text data. On one hand, due to the restrictive topics and coverage of a text corpus, some words might not have sufficient contexts and therefore might not have reliable word embeddings. On the other hand, even if a word has sufficient contextual data, the free texts might be inaccurate, thus might not provide a semantically precise view of the word. As a result, the learned word embedding might be unable to carry on the desirable semantic information. To tackle this problem, recently some researchers have proposed to leverage knowledge graphs, such as WordNet [17] and Freebase [3], as additional data sources to improve word embedding [2; 21; 23].

A knowledge graph contains a set of nodes representing entities and a set of edges corresponding to the relationships between entities. In other words, a knowledge graph can be regarded as a set of triples (h, r, t) , where head entity h and tail entity t share the relationship r . In [23; 21], in addition to the original likelihood loss on the free texts, an extra loss function is imposed to capture the relationships in the knowledge graph. Specifically, the additional loss takes the form $L_K = \sum_{(h,r,t)} \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$, where \mathbf{h} , \mathbf{t} are the embedding vectors of the words (entities) h and t respectively, and \mathbf{r} is the embedding of the relationship r .¹ Then the embeddings are learned by minimizing the overall loss on both free text and knowledge graph.

While the above approaches have shown certain success, we would like to point out their limitations.

First, the loss function L_K in these works cannot capture complex relationships between entities. In particular, it will encounter problem when the relationships are *one-to-many*, *many-to-one*, or *many-to-many*. For example, $r = \text{“cause of death”}$ is a *many-to-one* relationship, since many different head entities h_i (e.g., $h_1 = \text{“Abraham Lincoln”}$ and $h_2 = \text{“John F Kennedy”}$) correspond to the same tail entity (e.g., $t = \text{“assassination by firearm”}$). In this case, the minimization of L_K will enforce the embedding vectors of all head entities (e.g., $\mathbf{h}_1, \mathbf{h}_2$) to approach each other, which is clearly unreasonable.

Actually such kind of complex relationships are very com-

¹Please note that throughout the paper we will use bold characters to represent the embedding vectors for corresponding items.

mon in knowledge graphs. Take a widely used benchmark dataset *FB13* [19], which is a subset of Freebase, as an instance. For every relationship in *FB13*, we calculate the average number of head entities corresponding to one tail entity and the average number of tail entities corresponding to one head entity. Then we obtain the means and standard deviations of such values under different relationships. The overall statistical information is listed in Table 1, from which we can see that relationships in *FB13* are highly non *one-to-one*, especially for the mapping from tail entity to head entity, as shown by the large mean value of *#Head per Tail*. In addition, the standard deviation for *#Head per Tail* is fairly large, indicating that the degrees of non *one-to-one* mappings from tail entity to head entity vary drastically across different relationships. This clearly shows that the issue is very serious and we should tackle it in order to learn a reasonable word embedding.

#Tail per Head		#Head per Tail	
Mean	Std.Deviation	Mean	Std.Deviation
1.26	0.23	2614.17	9229.75

Table 1: Number of head entities per tail entity and number of tail entities per head entity in *FB13*.

Second, the loss function L_K adopts simple arithmetic operations on the embedding vectors of the head and tail entities, implying that both entities are located in the same space. However, the fact is that head entities are usually more concrete and tail entities are more abstract, making it unreasonable to simply regard them as in a homogeneous space. Still use the above example, for the relationship r = “cause of death”, all the head entities are real human names whereas all the tail entities are abstract reasons of death. What’s more, according to Table 1, head and tail entities are not symmetric from the statistics perspective: the number of tail entities per head entity is much smaller than that of head entities per tail entity, further indicating the heterogeneity nature of head and tail entities and suggesting that we should treat them separately in the mathematical modeling.

In the literature, there are some research works that try to resolve one of the aforementioned issues, however, as far as we know, none of the works successfully addressed both issues. For example, in [22], it is proposed to project the embedding vectors of both entities onto a relation-dependent hyperplane before computing the loss function L_K . However, the heterogeneity between head and tail entities is not considered. Furthermore, the projection matrix used in [22] has a fixed rank for all types of relationships, which could not express various degrees of non *one-to-one* mappings. In [5], different transformations are adopted to head and tail entities respectively, however, no consideration is taken to address the issue of non *one-to-one* mappings.

To address the limitations of existing works, in this paper, we propose a new algorithm called ProjectNet, which adopts different and carefully designed projections to the head and tail entities respectively when defining the loss function L_K . First, we show that the necessary condition to resolve the issue of non *one-to-one* mapping is to ensure the projection matrix to be low-rank. In such a way, we can guarantee the trans-

lation distance between the entities to be small after projection without forcing their embedding vectors to be the same. Actually, it can be proven that the TransH model in [22] is our special case in the sense that it also adopts a projection matrix of low (and fixed) rank. Our model is more general since we can explicitly control the rank of the projection matrix, so as to adapt to knowledge graphs with different degrees of non *one-to-one* mappings. Second, by using different projection matrices for head and tail entities respectively, we can avoid the homogeneity assumption on the semantic space and therefore build a more flexible and accurate model. For example, for the knowledge graph *FB13*, we should adopt a low-rank projection matrix for head entities since the number of head entities is very large for each tail entity; however, it is safe to use a relatively full rank projection matrix for tail entities since the number of tail entities is rather small for each head entity.

We have tested the performance of our proposed algorithm on several benchmark datasets, and the experimental results show that our proposal can significantly outperform the baseline methods. This indicates the benefit of carefully modeling entities and relationships when incorporating knowledge graphs into the learning process of word embedding.

The rest of the paper is organized as following. In Section 2, we summarize related works in leveraging knowledge graph to help word embedding. Then in Section 3, the detailed model is introduced and its difference with related methods is illustrated. After that, the experimental settings and results are reported in Section 4. The paper is finally concluded in Section 5.

2 Related Work

Word embeddings, (a.k.a. distributed word representations) are usually trained with neural networks by maximizing the likelihood of a text corpus. Based on several pioneering efforts [1; 7; 20], the research works in this field have grown rapidly in recent years [11; 15; 16; 18; 6]. Among them, *word2vec* [15; 16] draws quite a lot of attention from the community due to its simplicity and effectiveness. An interesting result given by *word2vec* is that the word embedding vectors it produces can reflect human knowledge via some simple arithmetic operations, e.g., $v(\text{Japan}) - v(\text{Tokyo}) \approx v(\text{France}) - v(\text{Pairs})$.

However, as aforementioned, word embedding models like *word2vec* usually suffer from the incompleteness and inaccuracy of the free-text training corpus. To address this challenge, there are some attempts that leverage additional structured or unstructured human knowledge to enhance word embeddings. Here are some examples. In [14; 8], the authors adopted morphological knowledge to aid the learning of rare words and new words. In [24], the authors used semantic relational knowledge between words as a constraint in learning word embedding vectors. In [23], the authors leveraged knowledge graphs, the most widely used structured knowledge, to help improve word representations. In particular, the authors did not only minimize the loss on the text corpus, but also minimized the loss on the knowledge graph by sharing embedding vectors between words and entities. In

[21], the authors proposed a very similar method to [23], but with a different objective of improving knowledge graph understanding with the help of text corpus. Actually, both the models in [23] and [21] are inspired by the TransE model [4], which is a state-of-the-art work in the literature of computing distributed representations for knowledge graphs [5; 12; 19]. In TransE, the relational operation between entities h, t with relationship r is assumed to be a simple linear translation, i.e., $\min \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. However, as pointed out in the introduction, such a simple formulation cannot handle the non *one-to-one* mappings between entities. To tackle the problem, in [22], the authors proposed a simple projection method named TransH. We will review the detailed mathematical forms of these models in Section 3.3 and discuss their relationship with our proposal.

3 The ProjectNet Algorithm

In this section, we introduce our proposed ProjectNet model in details. In general, following [23; 21], given a training text corpus \mathcal{D} and a set \mathcal{K} of triples in the form (*head entity, relation, tail entity*) extracted from a knowledge graph, our model jointly minimize a linear combination of the loss items on both text and knowledge:

$$L = \alpha L_{\mathcal{D}} + (1 - \alpha) L_{\mathcal{K}}, \quad (1)$$

where $\alpha \in [0, 1]$ is used to trade off the two loss terms. $L_{\mathcal{D}}$ and $L_{\mathcal{K}}$ share the same parameters, i.e., the embedding vectors for words and their corresponding entities are the same. In the following subsections, we will introduce the text model to specify $L_{\mathcal{D}}$ and the knowledge model to specify $L_{\mathcal{K}}$.

3.1 Text Model

Similar to [23; 21], we leverage the Skip-Gram model [16] as the text model. In Skip-Gram, the probability of observing the target word w_O given its context word w_I is modeled as $P(w_O|w_I) = \frac{\exp(\mathbf{w}_O \cdot \mathbf{w}_I)}{\sum_{w \in \mathcal{V}} \exp(\mathbf{w}' \cdot \mathbf{w}_I)}$, where $\mathbf{w} \in \mathcal{R}^d$ and $\mathbf{w}' \in \mathcal{R}^d$ denote the input and output embedding vectors for word w respectively, \mathcal{V} is the dictionary, and d is dimension of the embedding.

Given the training corpus \mathcal{D} consisting of $|\mathcal{D}|$ token words $\{p_1, \dots, p_k, \dots, p_{|\mathcal{D}|}\}$, the loss $L_{\mathcal{D}}$ is specified by:

$$L_{\mathcal{D}} = \sum_{k=1}^{|\mathcal{D}|} \sum_{j \in \{-M, \dots, M\}, j \neq 0} P(p_k | p_{k+j}), \quad (2)$$

where $2M$ is the size of the sliding window. As it is expensive to directly minimize $L_{\mathcal{D}}$ due to the denominator of $P(w_O|w_I)$, we adopt the negative sampling strategy [16] to boost the computation efficiency.

3.2 Knowledge Model

The knowledge model in ProjectNet is based on an *asymmetric low-rank projection* that projects the original entity embedding vectors into a new semantic space. The projection is designed to be *asymmetric* in order to handle the heterogeneity between head and tail entities, and is designed to be *low-rank* in order to deal with non *one-to-one* relationships in the knowledge graphs.

Asymmetric Projection

As aforementioned, the head and tail entities in knowledge graphs are usually very different, from both semantic and statistical perspectives. Therefore, we argue that it is unreasonable to adopt the same projection to these two kinds of entities (as TransH [22] does). Instead, it would be better to adopt different projection matrices, denoted as $L_r \in \mathcal{R}^{d \times d}$ and $R_r \in \mathcal{R}^{d \times d}$ respectively, to the head and tail entities. Hence, given a triple (h, r, t) , the original embedding vectors for h and t will be transformed to \mathbf{h}' and \mathbf{t}' as follows,

$$\mathbf{h}' = L_r \mathbf{h}, \quad \mathbf{t}' = R_r \mathbf{t}. \quad (3)$$

Based on the transformed embeddings, we define a scoring function f_d to reflect the confidence level that the triple (h, r, t) is true:

$$f_d(h, r, t) = \|\mathbf{h}' + \mathbf{r} - \mathbf{t}'\|_2^2 = \|L_r \mathbf{h} + \mathbf{r} - R_r \mathbf{t}\|_2^2. \quad (4)$$

Then we adopt a margin based ranking loss to distinguish the golden relationship triples from randomly corrupted triples:

$$L_{\mathcal{K}} = \sum_{(h,r,t)} \sum_{(h',r',t') \in N(h,r,t)} [\gamma - f_d(h', r', t') + f_d(h, r, t)]_+, \quad (5)$$

where $[x]_+ = \max(0, x)$, $\gamma > 0$ is the margin value, $N(h, r, t)$ is the set of all the corrupted triples built for the triple (h, r, t) , and L_r and R_r will be specified in (8).

Low-Rank Projection

As mentioned in the introduction, many relationships in the knowledge graphs are non *one-to-one*. In this case, in order to achieve reasonable results during the minimization of $L_{\mathcal{K}}$ defined above, it is necessary to constrain the projection matrices L_r and R_r to be *low-rank*, which is described in the following proposition.

Proposition 3.1 *Once linear projections are imposed to head and tail entities, the necessary condition to overcome the non one-to-one mapping problem is that the projection matrices L_r and R_r should not be full-ranked.*

Proof Consider the following least-square problem w.r.t. the optimization variable \mathbf{h} :

$$\min \|\mathbf{L}_r \mathbf{h} - \mathbf{c}\|_2^2, \quad (6)$$

where $\mathbf{L}_r \mathbf{h} = \mathbf{h}'$ and we regard $\mathbf{c} = \mathbf{t}' - \mathbf{r}$ as a constant vector. It is easy to obtain that the optimal solution \mathbf{h}^* satisfies the following linear system:

$$L_r^T L_r \mathbf{h}^* = L_r^T \mathbf{c}. \quad (7)$$

To avoid the non one-to-one mapping problem, the above equation must have multiple solutions. Then it is necessary that $L_r^T L_r$ is a low-rank matrix. In addition, as $\text{rank}(L_r^T L_r) = \text{rank}(L_r)$, the linear projection matrix L_r must not be full-rank either. The same conclusion holds for the projection matrix R_r for the tail entity. ■

Given the above proposition, we use the following tricks to ensure that L_r and R_r are low-rank matrices (whose ranks are m_L and m_R respectively, with $m_L < d$ and $m_R < d$):

$$L_r = \sum_{i=1}^{m_L} \mu_r^{(i)} \mathbf{p}_r^{(i)} \mathbf{q}_r^{(i)T}, \quad R_r = \sum_{i=1}^{m_R} \zeta_r^{(i)} \mathbf{o}_r^{(i)} \mathbf{s}_r^{(i)T}, \quad (8)$$

where $\mu_r^{(i)}$, $\zeta_r^{(i)}$ are scalars, and $\mathbf{p}_r^{(i)}$, $\mathbf{q}_r^{(i)}$, $\mathbf{o}_r^{(i)}$, $\mathbf{s}_r^{(i)}$ are all d -dimensional real vectors, the outer products of which constitute $(m_L + m_R)$ rank 1 matrices $\mathbf{p}_r^{(i)}\mathbf{q}_r^{(i)T}$ and $\mathbf{o}_r^{(i)}\mathbf{s}_r^{(i)T}$. For simplicity, we set the rank of all the left matrices L_r to be the same (m_L) and the rank of all the right matrices R_r to be the same (m_R). Please note we can also specify different ranks for different relationships r . We leave the corresponding discussions to the future work.

3.3 Discussions

In this section, we discuss the connections of our proposed ProjectNet algorithm with a few previous works and show that they are special cases of ProjectNet.

RNet. RNet refers to the knowledge models proposed in [23] and [21]. In fact, both models in the two works try to minimize the same scoring function: $f_d(h, r, t) = \|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_2^2$. Their only difference lies in how f_d is minimized. In [23], a large margin ranking loss is adopted for the minimization of $f_d(h, r, t)$, whereas in [21], an approximate softmax loss is used. It is clear that such a scoring function $f_d(h, r, t)$ cannot handle either the non *one-to-one* relationships between entities or the heterogeneity between head and tail entities. To state it more formally, let us consider the relationship triples (h_i, r, t) , $i \in 1, \dots, N$, where all head entities h_i have the same relationship r with tail entity t . In the ideal case, if all $f_d(h_i, r, t)$ are fully minimized, we will have $\mathbf{h}_i = \mathbf{t} - \mathbf{r}$, $\forall i \in 1, \dots, N$, which implies that $\mathbf{h}_1 = \mathbf{h}_2 = \dots = \mathbf{h}_N$. It means that all the embedding vectors for the head entities $\{h_i\}_{i=1}^N$ are the same, which is clearly unreasonable. We may encounter similar issues for *one-to-many* relationships $\{(h, r, t_j)\}_j$ and *many-to-many* relationships $\{(h_i, r, t_j)\}_{i,j}$.

Note that RNet corresponds to $L_r = R_r = I_{d \times d}$ in (3) and since the identity matrix $I_{d \times d}$ can be written in the form of (8), RNet can be regarded as a special case of ProjectNet.

TransH. TransH [22] is proposed to overcome the non *one-to-one* mapping problem. It first projects the entity embedding vectors \mathbf{h} and \mathbf{t} onto a hyperplane w.r.t the relationship r , and then the projected vectors \mathbf{h}_\perp and \mathbf{t}_\perp are used to define the scoring function f_d . Specifically,

$$\begin{aligned} \mathbf{h}_\perp &= \mathbf{h} - \mathbf{w}_r^T \mathbf{h} \mathbf{w}_r, \mathbf{t}_\perp = \mathbf{t} - \mathbf{w}_r^T \mathbf{t} \mathbf{w}_r, \\ f_d(h, r, t) &= \|\mathbf{h}_\perp + \mathbf{r} - \mathbf{t}_\perp\|_2^2, \end{aligned} \quad (9)$$

where $\mathbf{w}_r \in \mathcal{R}^d$ is the normal vector of the hyperplane with unit length (i.e., $\mathbf{w}_r \cdot \mathbf{r} = 0$ and $\|\mathbf{w}_r\|_2 = 1$).

Our proposed ProjectNet model differs from TransH in two ways: (i) we adopt different projections to head and tail entities; (ii) we adopt general projection matrices rather than a hyperplane based projection. Actually, TransH (9) can be regarded as a special case of ProjectNet (5), as shown below. Starting from (9), we have

$$\mathbf{h}_\perp = \mathbf{h} - \mathbf{w}_r^T \mathbf{h} \mathbf{w}_r = \mathbf{h} - \mathbf{w}_r \mathbf{w}_r^T \mathbf{h} = (I - \mathbf{w}_r \mathbf{w}_r^T) \mathbf{h}. \quad (10)$$

Hence, by substituting $L_r = (I - \mathbf{w}_r \mathbf{w}_r^T)$ in (3)(4), we get TransH. We still need to check whether $L_r = (I - \mathbf{w}_r \mathbf{w}_r^T)$ can be written in the form of (8), i.e., the weighted sum of

m_L rank-1 matrices, where $m_L < d$. We answer this question in the following two steps: (i) As $L_r = (I - \mathbf{w}_r \mathbf{w}_r^T)$ is an idempotent matrix (i.e. $L_r L_r = L_r$) and \mathbf{w}_r is a unit length vector, it holds that $\text{rank}(L_r) = \text{trace}(L_r) = d - 1$ [10]. Therefore, L_r has $d - 1$ non-zero eigenvalues. Furthermore, by observing that the eigenvalues of $\mathbf{w}_r \mathbf{w}_r^T$ are 0 and 1, we can conclude that $L_r = (I - \mathbf{w}_r \mathbf{w}_r^T)$ has 1 as one of its eigenvalues, corresponding to $d - 1$ linearly independent eigenvectors, and 0 as its another eigenvalue, corresponding to one eigenvector. (ii) Further considering that L_r is a real symmetric matrix, we can decompose L_r as $L_r = U_r \Sigma_r U_r^T$, where $U_r = (\mathbf{u}_r^{(1)}, \mathbf{u}_r^{(2)}, \dots, \mathbf{u}_r^{(d)}) \in \mathcal{R}^{d \times d}$ and $\Sigma_r = \text{diag}(1, 1, \dots, 1, 0) \in \mathcal{R}^{d \times d}$. The first $d - 1$ columns $\{\mathbf{u}_r^{(i)}\}_{i=1}^{d-1}$ of U_r are all the unit-length eigenvectors of L_r corresponding to eigenvalue 1 and Σ_r stores all the eigenvalues of L_r . Thus we can write $L_r = \sum_{i=1}^{d-1} \mathbf{u}_r^{(i)} \mathbf{u}_r^{(i)T}$.

The same procedure holds for the relation between R_r and \mathbf{t}_\perp . Then according to the above discussions, we can obtain the following proposition.

Proposition 3.2 *In the knowledge model of ProjectNet (8)(4), letting $m_L = m_R = d - 1$, $\mu_r^{(i)} = \zeta_r^{(i)} = 1$ and $\mathbf{p}_r^{(i)} = \mathbf{q}_r^{(i)} = \mathbf{o}_r^{(i)} = \mathbf{s}_r^{(i)} = \mathbf{u}_r^{(i)}$, where $\mathbf{u}_r^{(i)}$ is the i^{th} eigenvector of the matrix $I - \mathbf{w}_r \mathbf{w}_r^T$ with unit length, $i = 1, 2, \dots, d - 1$, we can obtain the TransH model (9).*

SE. SE [5] adopts the following scoring function:

$$f_d(h, r, t) = \|L_r \mathbf{h} - R_r \mathbf{t}\|_1. \quad (11)$$

SE looks very similar to our proposed knowledge model. However, there is a key difference: SE does not add the low-rank constraints to the matrices L_r and R_r . In other words, they fix the rank of these two matrices to be full, while in our model the rank of the matrices is a variable. Therefore our model is more general than SE and can handle the non *one-to-one* relationship when the rank is low while SE cannot since its rank is always full. In this sense, we could also regard SE as a special case of our proposed ProjectNet model.

TransR. [13] TransR treats relationships and entities as different objects and thus separates their embeddings into different spaces,

$$f_d(h, r, t) = \|M_r \mathbf{h} + \mathbf{r} - M_r \mathbf{t}\|_2^2. \quad (12)$$

Different with our formulation (8)(5), they did not add the low-rank constraint to matrix M_r (or we say that it sets the matrix M_r to be full-rank). In addition, TransR adopts the same transformation matrices to head and tail entities, by assuming that they are located in the same space. Therefore, the knowledge model in our ProjectNet algorithm is more general than TransR, and can include it as our special case.

4 Experiments

In this section, we conduct a set of experiments to verify the effectiveness of the ProjectNet model.

4.1 Experiments Setup

Training Data

For the free text corpus, we used a public snapshot of English Wikipedia named *enwik9*.² The corpus contains about 120

²<http://matmahoney.net/dc/enwik9.zip>

million word tokens. We removed digital words and words with frequency less than 5. Then we leveraged a knowledge graph *FB13* [19] to impose relationships onto those entities covered by *enwik9*. Since *FB13* contains many entities whose names have multiple words, in *enwik9* we merged these words into phrases and regarded both single words and phrases as embedding units in the dictionary. Finally the dictionary size is about 230k.

Baseline Methods

We consider the following algorithms as our baselines (we used the codes released by the authors of these works for implementation):

1. **Skip-Gram (SG)**: the original Skip-Gram model in *word2vec*, corresponding to $\alpha = 0$ in (1).
2. **RNet**: the joint embedding model in [23] and [21], which adopts the objective $\min ||\mathbf{h} + \mathbf{r} - \mathbf{t}||_2^2$ in the knowledge model.³
3. **Skip-Gram+TransH (SG+TransH)**: the combination of Skip-Gram (for the text model) and TransH (for the knowledge model). According to the discussions in Section 3.3, this baseline is a special case of ProjectNet.

Parameter Setting

In our experiments, we set the embedding size to $d = 100$. Stochastic Gradient Descent (SGD) is used to train all the models. We set the initial learning rate to be 0.025 and linearly dropped it during the training process. For the knowledge model in ProjectNet, we initialized the projection matrices L_r and R_r to be diagonal matrices with randomly assigned 0, 1 elements (with m_L and m_R non-zero elements respectively). For m_L and m_R , we varied their values according to the set $\{10, 20, \dots, 80, 90, 95, 100\}$. For all the joint embedding models, we varied the trade-off parameter α in (1) according to the set $\{0.01, 0.05, 0.1, 0.2, 0.5\}$. The margin value is set to $\gamma = 1$.

We used two tasks to evaluate our algorithm and the baseline models, one is the analogical reasoning task and the other is the word similarity task. The corresponding experiments results are shown in the following two subsections.

4.2 Analogical Reasoning Task

The analogical reasoning task is a word relationship inference task proposed in [16]. It consists of several quadruple word questions $a:b:c:d$, in which the relationship between word a and b is the same as that between c and d . For instance, $(a : b, c : d) = (\text{Berlin} : \text{Germany}, \text{Paris} : \text{France})$ and the relationship r is *capital-countries*. The task aims to infer word d given words a , b , and c using their word embedding vectors. To be more concrete, the inferred word \hat{d} is given by $\hat{d} = \arg \max_{w \in \mathcal{V}} \cos(\mathbf{b} - \mathbf{a} + \mathbf{c}, \mathbf{w})$. Once $\hat{d} = d$, the result on this quadruple word question is right; otherwise, it is wrong.

³As aforementioned, the models in the two papers differ only in the loss function (i.e., ranking loss vs. approximate softmax loss). Hence, we unify these two models using the name RNet and report the better performance of the two loss functions.

To construct the test set for the analogical reasoning task, we randomly sampled 1% triples from *FB13*, and filtered them according to the dictionary of *enwik9*.⁴ This test set consists of about 20k questions belonging to 7 non *one-to-one* relationships. The detailed statistics for this test dataset can be found in Table 2.

Then, we went through the remaining triples in *FB13* and removed all those triples containing overlapped entities with the test data. In this way, we obtained a training set with roughly 76k triples, which has no overlap with the test set in either relationship triples or entities. The goal of doing so is to examine whether the free text corpus can act as a bridge between known and unknown entities, so as to verify the necessity of jointly embedding text and knowledge into the distributed representation space.

For ProjectNet, as we imposed $L_r \mathbf{a} - R_r \mathbf{b} = -\mathbf{r} = L_r \mathbf{c} - R_r \mathbf{d}$ instead of $\mathbf{a} - \mathbf{b} = \mathbf{c} - \mathbf{d}$, we took a two-step approach instead of directly using the original word vectors to perform the analogical reasoning task: (i) we chose an optimal relationship r^* that best describes the relationship between a and b , i.e., $r^* = \arg \min_r ||L_r \mathbf{a} + \mathbf{r} - R_r \mathbf{b}||_2^2$; (ii) under r^* , we chose the answer word \hat{d} according to $\hat{d} = \arg \min_{w \in \mathcal{V}} ||L_{r^*} \mathbf{c} + \mathbf{r}^* - R_{r^*} \mathbf{w}||_2^2$. The same evaluation method was also applied to SG+TransH as well.

The experimental results are listed in Table 2, from which we have the following observations:

- All the knowledge based models (RNet, SG+TransH, and ProjectNet) outperform the original SG model, indicating that the quality of word embedding can be improved by leveraging knowledge graphs.
- The two models that take non *one-to-one* mappings into consideration (i.e., SG+TransH and ProjectNet) are superior to RNet, showing the necessity of modeling the non *one-to-one* mappings into the loss functions.
- Among all the models, ProjectNet achieves the best performance in all the seven subtasks. For the overall accuracy, it achieves over 30% relative gain than SG+TransH. This well demonstrates the advantages of our proposed model.

Sensitivity to different ranks

The best performance of ProjectNet was obtained with $\alpha = 0.2$, $m_L = 50$, and $m_R = 90$. To show the influence of the ranks of the projection matrices, in Figure 1, we plotted two curves that reflect the performance of ProjectNet w.r.t. different rank values m_L and m_R : one curve corresponds to changing m_R while fixing $m_L = 50$, and the other corresponds to changing m_L while fixing $m_R = 90$. From the figure, we have the following observations. (i) The performance becomes bad when the rank is too low. This is because in this case the model expressiveness becomes poor due to small number of free parameters in the projection matrices. (ii) For the projection matrix for head entities, medium values of m_L correspond to the better performances (the dashdot line),

⁴We did not use the analogical reasoning dataset given in [16] because this dataset is too special in the sense that almost all the relationships in it are *one-to-one* mappings.

Relationship	#Question	SG	RNet	SG+TransH	ProjectNet
<i>cause_of_death</i>	4290	3.29%	4.31%	7.55%	10.84%
<i>nationality</i>	870	14.60%	14.14%	14.82%	17.47%
<i>gender</i>	650	67.08%	59.38%	75.54%	84.62%
<i>profession</i>	6320	3.73%	5.78%	8.66%	13.42%
<i>institution</i>	4556	1.54%	3.03%	4.92%	6.72%
<i>ethnicity</i>	342	16.96%	15.50%	15.79%	18.71%
<i>religion</i>	3192	13.00%	12.47%	15.88%	22.06%
Total	20220	7.33%	8.15%	11.24%	15.28%

Table 2: Accuracy of different models on analogical reasoning task.

while for the projection matrix for tail entities, higher values of m_R lead to better performances (the solid line). This result is consistent with the statistical information in Table 1: the degree of non *one-to-one* mappings for head entities is much higher than that for tail entities, suggesting a lower rank of projection matrix for head entities.

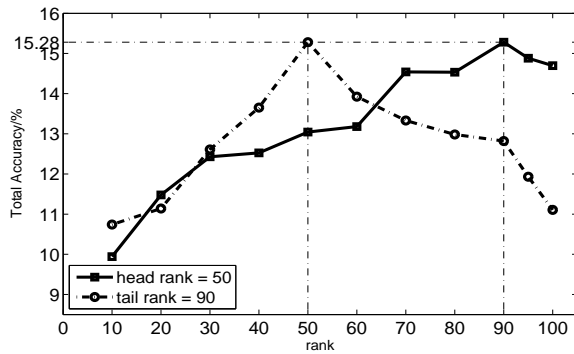


Figure 1: Accuracy w.r.t. different head and tail ranks. The dashdot line records the accuracy varying with different head ranks when tail rank is fixed to 90. The solid line records the accuracy varying with different tail ranks when head rank is fixed to 50.

4.3 Word Similarity Task

Word similarity is a task to investigate whether the similarity computed from word embedding vectors is consistent with human-labeled word similarity. We used three word-similarity tasks in our experiments, namely Word Similarity 353 (WS353) [9], SCWS [11] and Rare Word (RW) [14]. There are 353, 2003, and 2034 word pairs in these datasets respectively. From the word embedding vectors, we obtain the similarity scores (e.g., cosine similarity) for each word pair, based on which a ranked list is derived on the word pairs. Then the generated ranked list is compared to the ranked list produced by the ground-truth similarity scores assigned by human labelers. To evaluate the consistency between two ranking lists, we used Spearman’s Rank Correlation (denoted as $\rho \in [-1, 1]$). Higher ρ corresponds to better word embedding vectors.

Table 3 summarizes the results. For ProjectNet and SG+TransH, the word embedding vectors were directly used to compute the similarity scores, which is different from the analogical reasoning task. This is because there is no explicit relationship available in the evaluation process. The best performances of ProjectNet on the three datasets were obtained with the parameters setting to ($m_L = 50, m_R = 95, \alpha = 0.05$), ($m_L = 40, m_R = 90, \alpha = 0.01$), and ($m_L =$

$60, m_R = 95, \alpha = 0.05$) respectively. Table 3 reveals that ProjectNet achieves the best performance on all the datasets, which further indicates that ProjectNet produce higher quality word embedding vectors than the baseline methods.

Task/Model	SG	RNet	SG+TransH	ProjectNet
WS353	0.647	0.661	0.666	0.684
SCWS	0.610	0.614	0.618	0.630
RW	0.179	0.184	0.187	0.198

Table 3: Spearman’s Rank Correlation(ρ) on three Word Similarity Datasets: WS353, SCWS, and RW. Each ρ is reported as the average value of five repeated runs.

5 Conclusions and Future Work

In this paper we proposed a novel word embedding algorithm called ProjectNet, which leverages knowledge graphs to improve the quality of word embedding. In ProjectNet, we adopt different asymmetric low rank projections to head and tail entities in an entity-relationship triple, thus successfully maintain both non *one-to-one* mapping and heterogenous head/tail entities properties of knowledge graph. Experimental results demonstrate that ProjectNet significantly outperforms previous embedding models.

For the future work, we plan to apply the proposed approach to fulfill knowledge mining tasks, such as triplet classification and link prediction [22]. In addition, we plan to use the word embedding vectors generated by ProjectNet in some real-world applications such as document classification and web search ranking.

References

- [1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.
- [2] Jiang Bian, Bin Gao, and Tie-Yan Liu. Knowledge-powered deep learning for word embedding. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, pages 132–148, 2014.
- [3] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

- [4] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pages 2787–2795, 2013.
- [5] Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. Learning structured embeddings of knowledge bases. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.
- [6] Jan A. Botha and Phil Blunsom. Compositional Morphology for Word Representations and Language Modelling. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, Beijing, China, June 2014.
- [7] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.
- [8] Qing Cui, Bin Gao, Jiang Bian, Siyu Qiu, and Tie-Yan Liu. Learning effective word embedding using morphological word similarity. *CoRR*, abs/1407.1687, 2014.
- [9] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppín. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM, 2001.
- [10] R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.
- [11] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics, 2012.
- [12] Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems*, pages 3167–3175, 2012.
- [13] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [14] Minh-Thang Luong, Richard Socher, and C Manning. Better word representations with recursive neural networks for morphology. *CoNLL-2013*, 104, 2013.
- [15] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.
- [16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119, 2013.
- [17] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
- [18] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP 2014)*, 12, 2014.
- [19] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934, 2013.
- [20] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010.
- [21] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph and text jointly embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1591–1601. Association for Computational Linguistics, 2014.
- [22] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1112–1119, 2014.
- [23] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. Rc-net: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1219–1228. ACM, 2014.
- [24] Mo Yu and Mark Dredze. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 545–550, Baltimore, Maryland, June 2014. Association for Computational Linguistics.